NAME: CODRIN ONECI
SOURCES: 16.90 textbook

16.90 PSET 3 SOLUTION

PROBLEM 1

$$U_i = U_i$$
$$U_{i+1} = U_i + \Delta x U_{xi} + \frac{1}{2}\Delta x^2 U_{xxi} + \frac{1}{6}\Delta x^3 U_{xxxi} + O(\Delta x^4)$$
$$U_{i+2} = U_i + 2\Delta x U_{xi} + \frac{1}{2} 4\Delta x^2 U_{xxi} + \frac{1}{6} 8\Delta x^3 U_{xxxi} + O(\Delta x^4)$$

$$\frac{1}{\Delta x}(\alpha U_i + \beta U_{i+1} + \gamma U_{i+2}) = \frac{1}{\Delta x}(\alpha+\beta+\gamma)U_i + (\beta+2\gamma)U_{xi} + \frac{1}{2}(\beta+4\gamma)U_{xxi}\Delta x +$$
$$+ \frac{1}{6}(\beta+8\gamma)U_{xxxi}\Delta x^2 + O(\Delta x^3)$$

We can choose α, β, γ such that the first 3 terms are:

$$\left.\begin{aligned} \alpha+\beta+\gamma &= 0 \\ \beta+2\gamma &= 1 \\ \tfrac{1}{2}\beta+2\gamma &= 0 \end{aligned}\right\} \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & \frac{1}{2} & 2 \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow \boxed{\begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = \begin{pmatrix} -1.5 \\ 2 \\ -0.5 \end{pmatrix}}$$

Observe $\frac{1}{\Delta x}(-1.5U_i + 2U_{i+1} - 0.5U_{i+2}) - U_{xi} = -\frac{1}{3}U_{xxxi}\Delta x^2 + O(\Delta x^3)$ so this approximation is second order accurate.

PROBLEM 2

(a) $V>0 \rightarrow |V| = V \longrightarrow F(U_R, U_L) = \frac{V}{2} 2U_L = VU_L$

$$\Delta x \frac{dU_i}{dt} + VU_i - VU_{i-1} = 0 \longrightarrow \boxed{\frac{dU_i}{dt} + V\frac{U_i - U_{i-1}}{\Delta x} = 0}$$

This is equivalent to the backward space approximation finite difference discretization $U_{ix} \approx \delta_x^- U_i = \frac{U_i - U_{i-1}}{\Delta x}$, observing $\frac{\partial F}{\partial x} = VU_{xi}$.

(b) $$\Delta x\frac{dU_i}{dt} + F(U_{i+1}, U_i) - F(U_i, U_{i-1}) = \Delta x \frac{dU_i}{dt} + \frac{V}{2}(U_{i+1}+U_i) - \frac{V}{2}(U_i + U_{i-1}) =$$
$$= \boxed{\Delta x\frac{dU_i}{dt} + V\frac{U_{i+1} - U_{i-1}}{2}}$$

The equivalent finite difference approach uses $U_{xi} \approx \frac{U_{i+1} - U_{i-1}}{2\Delta x}$ and the corresponding discretization is:

$$\frac{dU_i}{dt} + V\frac{U_{i+1} - U_{i-1}}{2\Delta x} \approx \frac{\partial U}{\partial t} + V\frac{\partial U}{\partial x}$$

PROBLEM 3

a) $$\tau = U \frac{U_{i+1} - U_{i-1}}{2\Delta x} - \nu \frac{U_{i+1} - 2U_i + U_{i-1}}{\Delta x^2}$$

$$U_i = U_i ; \quad U_{i+1} = U_i + \Delta x U_{xi} + \frac{1}{2}\Delta x^2 U_{xxi} + \frac{1}{6}\Delta x^3 U_{xxxi} + \frac{1}{24}\Delta x^4 U_{xxxxi} + O(\Delta x^5)$$

$$U_{i-1} = U_i - \Delta x U_{xi} + \frac{1}{2}\Delta x^2 U_{xxi} - \frac{1}{6}\Delta x^3 U_{xxxi} + \frac{1}{24}\Delta x^4 U_{xxxxi} - O(\Delta x^5)$$

$$\frac{U_{i+1} - U_{i-1}}{2\Delta x} = \frac{2\Delta x U_{xi} + \frac{1}{3}\Delta x^3 U_{xxxi}}{2\Delta x} = U_{xi} + \frac{1}{6}\Delta x^2 U_{xxxi}$$

$$U_{i+1} - 2U_i + U_{i-1} = 2 \cdot \frac{1}{2}\Delta x^2 U_{xxi} + 2 \cdot \frac{1}{24}\Delta x^4 U_{xxxxi} + O(\Delta x^5)$$

$$\frac{U_{i+1} - 2U_i + U_{i-1}}{\Delta x^2} = U_{xxi} + \frac{1}{12}\Delta x^2 U_{xxxxi}$$

$$\tau = U\left(U_{xi} + \frac{1}{6}\Delta x^2 U_{xxxi}\right) - \nu\left(U_{xxi} + \frac{1}{12}\Delta x^2 U_{xxxxi}\right)$$

Observe $U U_{xi} - \nu U_{xxi} = 0$ because $U \frac{\partial U}{\partial x} = \nu \frac{\partial^2 U}{\partial x^2}$ was given.

$$\boxed{\tau = \Delta x^2 \left(\frac{U}{6} U_{xxxi} - \frac{\nu}{12} U_{xxxxi} \right)}$$

The local error suggests that this scheme is ~~se~~ FIRST order accurate.

b) Write $$U \frac{U_{i+1} - U_{i-1}}{2\Delta x} - \nu \frac{U_{i+1} - 2U_i + U_{i-1}}{\Delta x^2} = 0$$

$0 \quad 1 \quad \dots \quad N_{x-1} \quad N_x$

Dirichlet boundary — Dirichlet Boundary

Divide the set $[0, 1]$ into N_x equal cardinality compact sets.
The state vector is $U_{bc} = (U_0, U_1, \dots U_{N_x})^T$ but we know $U_0 = 1$ and $U_{N_x} = 0$ in all times, so I create $U = (U_1, \dots U_{N_x-1})^T$ that contains variable data. For $i \in \overline{1, N_{x-1}}$ we impose the discretized ODE so we can create an (N_x-1, N_{x-1}) matrix A with entries resulting from

$$\left(\frac{u}{2\Delta x} + \frac{\nu}{\Delta x^2}\right) u_{i-1} + \left(\frac{-2\nu}{\Delta x^2}\right) u_i + \left(\frac{-u}{2\Delta x} + \frac{\nu}{\Delta x^2}\right) u_{i+1} = 0$$

So $$\boxed{A_{i,i-1} = -\frac{u}{2\Delta x} - \frac{\nu}{\Delta x^2} ; \quad A_{i,i} = \frac{2\nu}{\Delta x^2} ; \quad A_{i,i+1} = \frac{u}{2\Delta x} - \frac{\nu}{\Delta x^2}}$$

We will write $Ax = b$ with $b_i = 0$ for $i = \overline{1, N_x-1}$

The boundary conditions must be incorporated into b_0 and b_{Nx}:

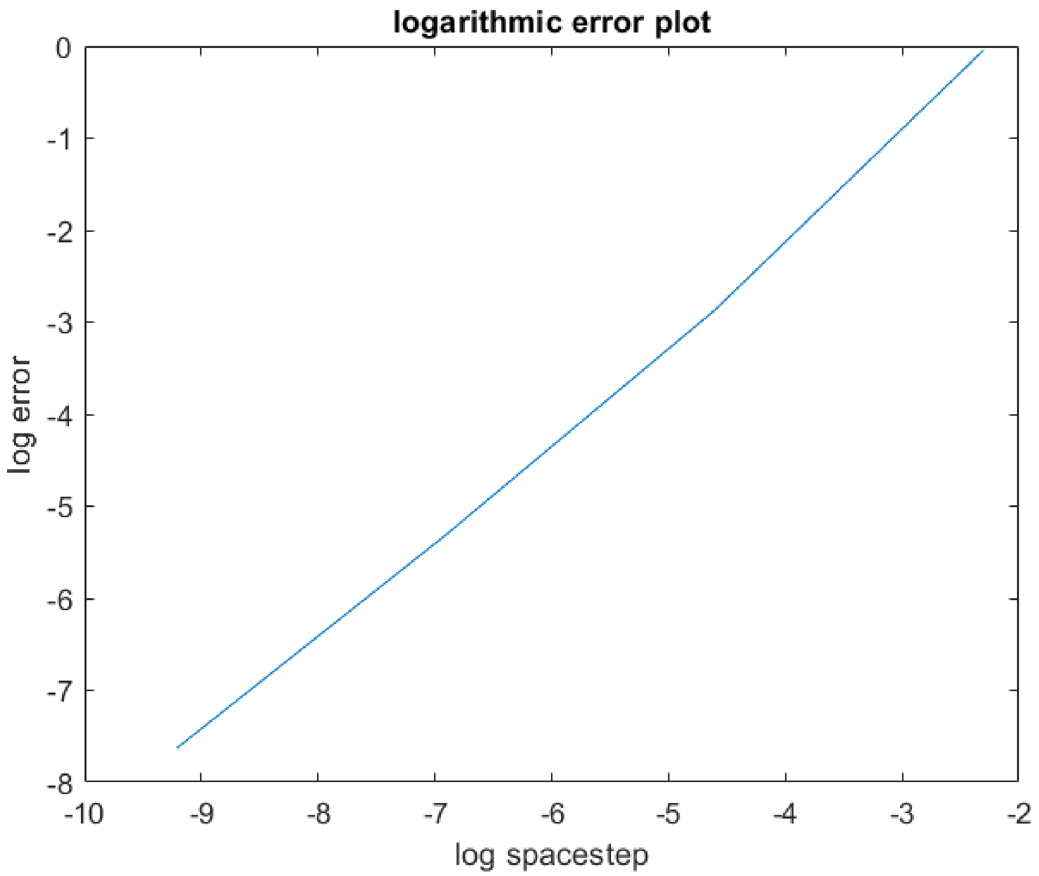
$$b_1 = \left(-\frac{u}{2\Delta x} - \frac{\nu}{\Delta x^2}\right)\underset{\overset{\uparrow}{1}}{U_0}\ ,\quad b_{Nx-1} = \left(\frac{u}{2\Delta x} - \frac{\nu}{\Delta x^2}\right)\underset{\overset{\uparrow}{0}}{U_{Nx-1}}$$

c) Put $N_x = [1/\Delta x]$, where $[\alpha] \equiv$ integer part of α

Even tough is not the most efficient method, we may create matrix A in the memory and then $u = A^{-1} b$ (A^{-1} was found with Gaussian elimination)

Acknowledgement: Thanks to the 16.90 TAs, I have identified the initial bug in my code. That was caused by not paying enough attention to MATLAB syntax. The initial statement was exact_u=zeros(Nx+1) and was generating a square matrix! I modified that statement into exact_u=zeros(1,Nx+1); and everything worked fine.

I have used the following code to generate the log-log plot:



As expected the slope is 1 so the method is first order accurate in space.

My code:

```
conv_velocity=1;
diffusivity=0.1;
c=1/(1-exp(conv_velocity/diffusivity));
%define Dirichlet boundary conditions
U_left_bc=1;U_right_bc=0;
dxs=[1/10, 1/100, 1/1000, 1/10000];
errs=zeros(1,4);
%errs=[1.5759,10.1571,31.6732,100.0159] values I found after running
code
for j=1:4
    dx=dxs(j);
    Nx=round(1/dx+1);
    A=zeros(Nx-1,Nx-1);
    for i=1:Nx-1
        if i==1
            A(i,i)=2*diffusivity/(dx^2);
```

```
            A(i,i+1)=conv_velocity/(2*dx)-diffusivity/(dx^2);
        elseif i==Nx-1
            A(i,i-1)=-(conv_velocity/(2*dx))-(diffusivity/(dx^2));
            A(i,i)=2*diffusivity/(dx^2);
        else
        A(i,i-1)=-(conv_velocity/(2*dx))-(diffusivity/(dx^2));
        A(i,i)=2*diffusivity/(dx^2);
        A(i,i+1)=conv_velocity/(2*dx)-diffusivity/(dx^2);
        end
    end
    b=zeros(Nx-1,1);
    b(1,1)=U_left_bc*(conv_velocity/(2*dx)+diffusivity/(dx^2));
    b(Nx-1,1)=-U_right_bc*(conv_velocity/(2*dx)-diffusivity/(dx^2));
    U_solution=A\b;
    exact_u=zeros(1,Nx+1);
    for i=1:length(exact_u)
        exact_u(i)=c*exp(i*dx*conv_velocity/diffusivity)+(1-c);
    end
       errs(j)=norm(exact_u-
[1,transpose(U_solution),0])/norm(exact_u);
end
disp(size(exact_u))
disp(size(U_solution))
%figure(1);
%plot(linspace(0,1,Nx+1),[1,transpose(U_solution),0],'-
',linspace(0,1,Nx+1),exact_u,'-')
%legend('State convection','exact');
%title('Convection');
%xlabel('location');
%ylabel('State');
disp(errs)
figure(2);
plot(log(dxs),log(errs),'-')
title('logarithmic error plot');
xlabel('log spacestep');
ylabel('log error');
```